IN THE CLAIMS:

Please Amend the claims to read as follows:

Please cancel claims 7-14, 16-19, and 33 without prejudice.

1. (Previously Presented) A method for converting a file access data structure from a first endianness to a second endianness by a processor, the method comprising the steps of:

identifying, from a descriptor look up table, a series of actions to perform on elements of the file access data structure, where the series of actions include at least one of converting, copying, or linking; and

performing the identified series of actions on the elements of the file access data structure to convert the file data structure from the first endianness to the second endianness.

2. (Previously Presented) A method of converting elements of a file access data structure from a first endianness to a second endianness by a processor, the method comprising the steps of:

determining if the file access data structure is a critical path data structure;

converting, in response to the file access data structure being a critical path data structure, the elements from the first endianness to the second endianness using a set of specific code functions;

converting, in response to the file access data structure not being a critical path data structure, a header of the file access data structure from the first endianness to the second endianness using a second set of specific code functions; and

calling a byte swapping engine to convert selected elements of the file access data structure from the first byte order to the second byte order.

3. (Original) The method of claim 2 wherein the file access data structure further comprises a direct access file access data structure.

4. (Previously Presented) A file system for converting elements of a file access data structure from a first endianness to a second endianness, the system comprising:

an input buffer, the input buffer storing the file access data structure with the first endianness to be converted;

a byte swapping engine, the byte swapping engine operative interconnected with a descriptor table, with the descriptor table listing a series of actions to perform when converting the file data structure from the first endianness to the second endianness, where the series of actions include at least one of converting, copying, or linking; and

an output buffer, the byte swapping engine placing the file access data structure with the second endianness in the output buffer after conversion.

5. (Original) The system of claim 4 wherein the descriptor table further comprises a set of entries describing various file access data structures, each entry further comprising a size field and an operation field.

6. (Original) The system of claim 4 wherein the file access data structure further comprises a direct access file access data structure.

7-14 (Cancelled)

15. (Previously Presented) A computer-readable medium, including program instructions executing on a computer, for converting elements of a file access data structure from a first endianness to a second endianness, the method comprising the steps of:

determining if the file access data structure is a critical path data structure;

converting, in response to the file access data structure being a critical path data structure, the elements from the first endianness to the second endianness using a set of specific code functions;

converting, in response to the file access data structure not being a critical path data structure, a header of the file access data structure from the first endianness to the second endianness using a second set of specific code functions; and

calling a byte swapping engine to convert selected elements of the file access data structure from the first byte order to the second byte order.

16-19 (Cancelled)

20. (Previously Presented) A method for converting a data structure by a processor, comprising:

calling a byte-swapping engine;

providing a file access data structure as input to the byte-swapping engine;

providing a descriptor look up table to the byte-swapping engine;

identifying, from the descriptor look up table, a series of actions to perform on elements of the file access data structure in order to swap bytes of the file access data structure from a first endianness to a second endianness, where the series of actions include at least one of converting, copying, or linking; and

performing the identified series of actions on the elements of the file access data structure to convert the file access data structure.

21. (Previously Presented) The method as in claim 20, further comprising:

using as the file access data structure a file having Direct Access File System (DAFS) protocol.

22. (Previously Presented) The method as in claim 20, further comprising:

determining if the file access data structure is a critical path data structure, where the critical path data structure includes commonly utilized data structures, and if the file

access data structure is a critical path data structure, perform byte swap operations using specific code functions.

23. (Previously Presented) The method as in claim 20, further comprising:

determining if the file access data structure is a critical path data structure, where the critical path data structure includes commonly utilized data structures, and if the file access data structure is not a critical path data structure, perform byte swap operations on a data structure header.

24. (Previously Presented) The method as in claim 20, further comprising:

swapping bytes of the data structure as needed, in response to swapping bytes of the file access data structure.

25. (Previously Presented) The method as in claim 20, further comprising:

determining if an element entry of the descriptor look up table is nested;

branching to the nested entry;

identifying, from the descriptor look up table, a nested series of actions to perform on elements of the nested entry in order to swap bytes of the entry from a first endianness to a second endianness, where the nested series of actions includes linking and converting.

26. (Previously Presented) A computer to convert a data structure by a processor, comprising:

means for calling a byte-swapping engine;

means for providing a file access data structure as input to the byte-swapping engine;

means for providing a descriptor look up table to the byte-swapping engine;

means for identifying, from the descriptor look up table, a series of actions to perform on elements of the file access data structure in order to swap bytes of the file access data structure from a first endianness to a second endianness, where the series of actions include at least one of converting, copying, or linking; and

means for performing the identified series of actions on the elements of the file access data structure to convert the file access data structure.

27. (Previously Presented) The computer as in claim 26, further comprising:

means for using as the file access data structure a file having Direct Access File System (DAFS) protocol.

28. (Previously Presented) The computer as in claim 26, further comprising:

means for determining if the file access data structure is a critical path data structure, where the critical path data structure includes commonly utilized data structures, and if the file access data structure is a critical path data structure, perform byte swap operations using specific code functions.

29. (Previously Presented) The computer as in claim 26, further comprising:

means for determining if the file access data structure is a critical path data structure, where the critical path data structure includes commonly utilized data structures, and if the file access data structure is not a critical path data structure, perform byte swap operations on a data structure header.

30. (Previously Presented) The computer as in claim 26, further comprising:

means for swapping bytes of the data structure as needed, in response to swapping bytes of the file access data structure.

31. (Previously Presented) The computer as in claim 26, further comprising:

means for determining if an element entry of the descriptor look up table is nested;

means for branching to the nested entry;

means for identifying, from the descriptor look up table, a nested series of actions to perform on elements of the nested entry in order to swap bytes of the entry from a first endianness to a second endianness, where the nested series of actions includes converting and linking.

32. (Previously Presented) A computer readable media, comprising:

said computer readable media containing instructions for execution on a processor for the practice of a method for converting a data structure by a processor, the method having the steps of,

calling a byte-swapping engine;

providing a file access data structure as input to the byte-swapping engine;

providing a descriptor look up table to the byte-swapping engine;

identifying, from the descriptor look up table, a series of actions to perform on elements of the file access data structure in order to swap bytes of the file access data structure from a first endianness to a second endianness, where the series of actions include at least one of converting, copying, or linking; and

performing the identified series of actions on the elements of the file access data structure to convert the file access data structure.

33. (Cancelled)

34. (Previously Presented) A method of converting elements of a file access data structure from a first endianness to a second endianness by a processor, comprising:

determining if the file access data structure is a critical path data structure; and

converting the elements from the first endianness to the second endianness using a set of specific code functions if the file access data structure is a critical path data structure.

35. (Previously Presented) The method of claim 34, further comprising:

converting a header of the file access data structure from the first endianness to the second endianness using a second set of specific code functions if the file access data structure is not a critical path data structure.

36. (Previously Presented) The method of claim 34, further comprising:

calling a byte swapping engine to convert selected elements of the file access data structure from the first byte order to the second byte order.

37. (Previously Presented) A method for converting a first data structure from a to a second data structure by a processor, the method comprising the steps of:

using a descriptor lookup table to provide actions to be performed on each element of the first data structure; and

stepping through the descriptor table and processing each element of the first data structure according to the element's size and action to convert the first data structure into the second data structure.

38. (Previously Presented) The method of claim 37, further comprising:

using a byte as the data structure.

39. (Previously Presented) The method of claim 2, wherein the critical data path structure includes commonly used data structures.

40. (Previously Presented) The method of claim 2, wherein the critical data path structure is a direct access file system (DAFS) header data structure.

41. (Previously Presented) The method of claim 2, wherein the specific code functions are designed to rapidly convert any elements of the data structure to the second endianness without using a byte swapping engine.

42. (Previously Presented) The computer-readable medium of claim 15, wherein the critical data path structure includes commonly used data structures.

43. (Previously Presented) The computer-readable medium of claim 15, wherein the critical data path structure is a direct access file system (DAFS) header data structure.

44. (Previously Presented) The computer-readable medium of claim 15, wherein the specific code functions are designed to rapidly convert any elements of the data structure to the second endianness without using a byte swapping engine.

45. (Previously Presented) The method of claim 34, wherein the critical data path structure includes commonly used data structures.

46. (Previously Presented) The method of claim 34, wherein the critical data path structure is a direct access file system (DAFS) header data structure.

47. (Previously Presented) The method of claim 34, wherein the specific code functions are designed to rapidly convert any elements of the data structure to the second endianness without using a byte swapping engine.